Engine RUL Prediction: Machine Learning Models Performance Comparison Study

Jack P. Lu
jlu455@gatech.edu

Abstract **—** Accurate estimation of Remaining Useful Life (RUL) is critical for predictive maintenance and risk mitigation in engineered systems. This project originally aimed to benchmark machine learning models for RUL prediction across all four NASA C-MAPSS turbofan datasets (FD001–FD004), utilizing a standardized workflow that integrated data preprocessing, piecewise-linear RUL labeling, normalization, sliding-window feature engineering, and model-agnostic feature selection via a genetic algorithm (GA). However, due to substantial computing and time constraints, the final evaluation was limited to FD001, as detailed in the Limitations section.

Four modeling approaches, Ridge Regression, Random Forest, XGBoost, and one-dimensional convolutional neural networks (1D-CNN), were systematically trained and compared using root mean squared error (RMSE) under both capped and true RUL targets. The results indicate that XGBoost achieved the strongest end-of-life prediction accuracy, with Random Forest and Ridge Regression yielding comparable performance, while 1D-CNN did not surpass the tree-based models on this benchmark. However, 1D-CNN demonstrated better generalization to the end-of-life prediction scenario than its average performance across all training windows might suggest.

These findings underscore the value of a model-agnostic, GA-based feature selection pipeline for fair model comparison and highlight the importance of rigorous, standardized preprocessing in prognostics research. Methodological reflections suggest that rapid prototyping and iterative development confer advantages over traditional linear workflows. Future work will extend this analytical framework to all C-MAPSS datasets to comprehensively evaluate model robustness and generalizability in more complex operational scenarios.

Table of Contents

INTRODUCTION

Accurate prediction of the Remaining Useful Life (RUL) of aging hardware components is critical for implementing effective predictive maintenance strategies, reducing downtime, and minimizing operational costs and risks. Leveraging NASA's Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) datasets, extensive research efforts have traditionally emphasized incremental accuracy improvements through increasingly complex modeling techniques. However, existing studies leveraging the C-MAPSS datasets have frequently depended on manually engineered or model-specific feature selection, which reduces generalization and hinders standardized comparison across methods (Ferreira & Gonçalves, 2022; Lei et al., 2018). Such non-robust approaches introduce inherent bias across models and evaluation inconsistencies, restricting the comparability of different predictive models (Hu et al., 2012). In response to these gaps, there is a growing expectation for a model agnostic pipeline to replace the handcrafted feature selection process that is highly dependent on SME advice and labor

resources which may introduce biases (Wu et al. 2024). To address these gaps, this project adopts a genetic algorithm (GA)–based, model-agnostic feature selection pipeline and systematically compares traditional machine learning methods against more complex machine learning models on the C-MAPSS benchmark dataset.

PROBLEM STATEMENT

This project seeks to evaluate the predictive performance of a range of machine learning models for estimating the Remaining Useful Life (RUL) of turbine engines, using NASA's Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) datasets.

Building on a foundation of extensive prior research that has already optimized numerous modeling approaches for this benchmark, the project's primary focus shifts towards a comparison analysis. Specifically, it compares traditional machine learning models such as Ridge Regression with more complex approaches, including ensemble methods like Random Forest and XGBoost, as well as deep learning model, one-dimensional convolutional neural networks (1D-CNN). The central question driving this study is whether advanced deep learning models offer significant performance gains over traditional models, thereby justifying their increased complexity.

LITERATURE SURVEY

Manual and Subject-matter Expert (SME)-driven feature selection, while foundational to the early development of prognostics and RUL modeling, faces significant limitations when addressing the complex and dynamic environments encountered in real-world operations. Traditional expert-driven approaches are typically anchored to known degradation modes, making them highly context-dependent and often unable to generalize across diverse fleets, operating conditions, or environmental conditions (Ramasso & Saxena, 2014). Industry experience supports this as well; a Boeing design engineer noted that degradation patterns for turbine engines are complex as there are multiple contextual factors to be considered, engines exposed to saltwater or intense sun require attention to different failure modes than those operating in more temperate or controlled environments (Design Engineer, personal communication, June 10, 2025). This underscores the limitations of static, ad-hoc approaches, as manual selection can overlook relevant features that emerge only under specific or changing operational profiles, and such processes may need to be revised repeatedly as deployment contexts shift.

To address these challenges, this project employs a model-agnostic, data-driven feature selection process using GA. By leveraging empirical validation rather than fixed expert

advice, GA-based selection offers a systematic and reproducible means to identify the most informative features for prediction, regardless of the modeling technique employed (Vafaie & Imam, 1994). GA-based feature selection enables fair and robust comparisons across a suite of machine learning models, such as Ridge Regression, Random Forest, XGBoost, and 1D-CNN, by ensuring each model is evaluated using the most predictive and unbiased subset of features. Unlike manual or model-specific selection methods, which may introduce subjective bias or favor certain algorithms, GA efficiently explores the feature space to maximize predictive accuracy according to the selected performance metric (Khumprom et al., 2020). Recent comparative studies further highlight that evolutionary search heuristics such as GA can meaningfully improve model reliability and generalizability, particularly in high-dimensional and industrial contexts (Safavi et al., 2024; Wang et al., 2023; Ensarioğlu et al., 2023).

The life cycle of a turbofan engine is segmented into distinct operational phases: normal operation with routine maintenance, degradation stage, and the eventual failure that marks the end of life for the engine. During the initial phase of normal operation, the bearing remains in a stable state, making RUL prediction unreliable and impractical. It is only after degradation begins that meaningful RUL prediction becomes feasible. (Li et al., 2025). To align with emerging best practices, this project adopts a two-stage piecewise linear degradation labeling approach, wherein the RUL remains constant during normal operation and begins to decay linearly after a data-driven inflection point. This methodology has been shown to generate more realistic and effective target labels for training deep learning models, addressing the complexities of nonlinear degradation processes in turbofan engines (Asif et al., 2022; Ensarioğlu et al., 2023).

To enhance the ability of learning algorithms to capture temporal degradation, a sliding time window methodology is employed. Unlike traditional rolling-window statistics, sliding windows segment the sensor data into sequential, forward-looking intervals, explicitly representing dynamic health trajectories and enabling models to learn complex temporal dependencies (Ensarioğlu et al., 2023; Wang et al., 2023).

Collectively, these studies highlight the critical need for systematic, model-agnostic feature selection and standardized preprocessing pipelines in modern RUL prediction. By aligning the project methodology with these best practices, our work seeks to establish a fair and transparent basis for comparing traditional and more complex models, while acknowledging the ongoing need to balance generalizability with computational cost and domain knowledge.

METHODOLOGY

Note: *This report presents a full methodology applicable to all four C-MAPSS datasets; however, feature selection, model training, and evaluation are completed on FD001 only. See Limitations for a discussion of the iterative development process and scope reduction rationale.*

Figure 1 illustrates the workflow of the proposed method, which is organized into five primary segments: data preprocessing, feature engineering, feature selection, model building and training, and testing and evaluation. The pipeline begins by importing and loading the C-MAPSS datasets. Raw data are cleaned and normalized in the preprocessing step, followed by sliding-time-window feature engineering and GA-based feature selection. The selected features are then used to train four machine learning models (see Figure 1). Model evaluation is conducted on the test data, using root mean squared error (RMSE) as the primary metric and assessing prediction significance via 10-fold cross-validation (CV). The flowchart concludes with documentation of findings, completing the entire workflow.

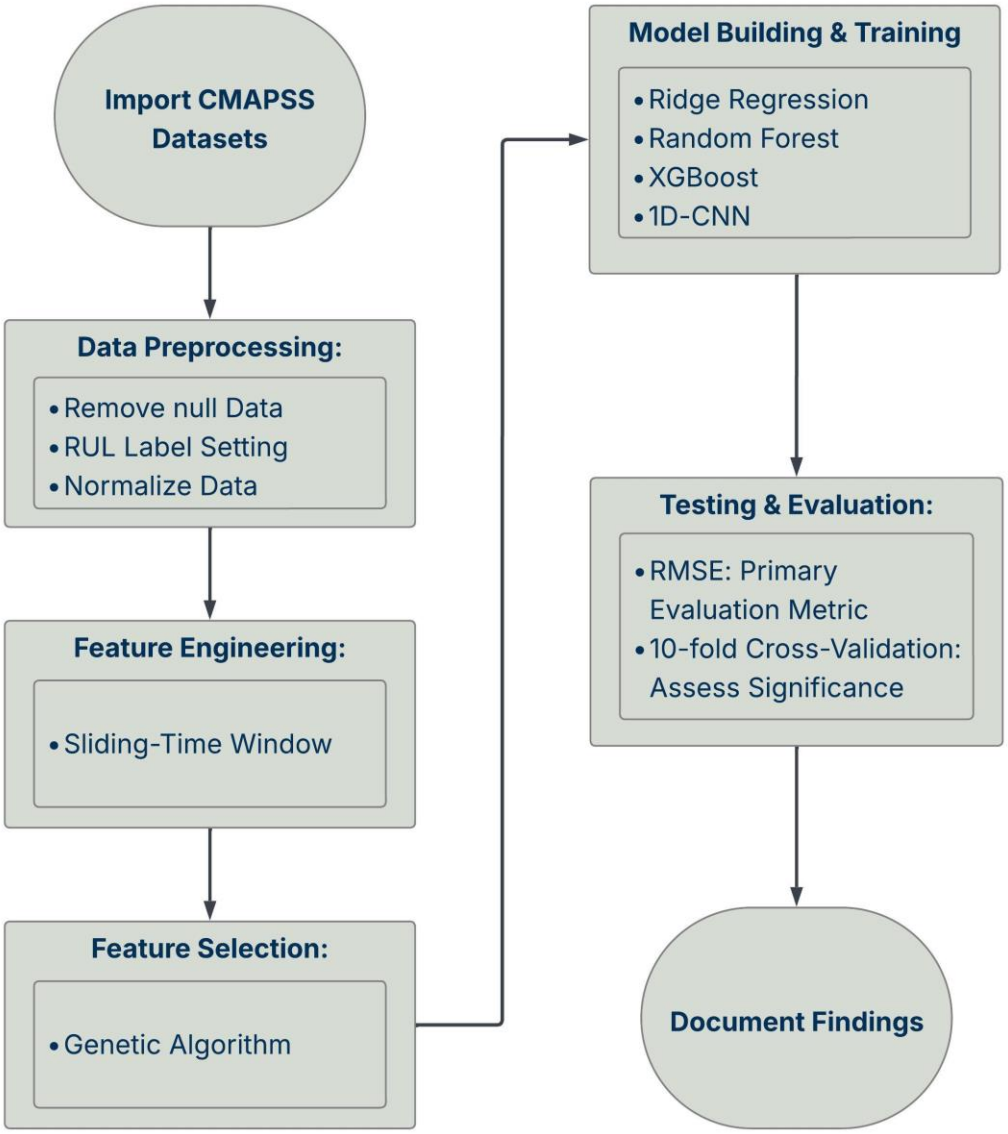

Figure 1. Flowchart illustrating the workflow of the proposed method.

Preliminary Exploratory Data Analysis

Preliminary Exploratory Data Analysis (EDA) was performed on the training partitions of all four C-MAPSS datasets (FD001–FD004) to assess data characteristics, sensor utility, the presence of missing values, and variable correlations before proceeding to data preprocessing and model development.

Data Characteristics

This study utilizes the NASA Turbofan Engine Degradation Simulation dataset (Saxena & Goebel, 2008), recognized as a benchmark for prognostics research. Table 1 summarizes the structure of the C-MAPSS datasets: FD001 and FD003 each contain 100 training and 100 test units under a single operating condition, with FD001 limited to one fault mode, High Pressure Compressor (HPC) degradation, and FD003 introducing dual fault modes (HPC and fan). FD002 and FD004 increase complexity, containing 260/259 and 248/249 train/test units, respectively, across six operating conditions and one (FD002) or two (FD004) faults. Each time series includes cycle-by-cycle engine IDs, three operational settings, and 21 anonymized sensor measurements. Training sequences run to failure, supporting RUL computation, while test sequences are truncated before failure, with ground-truth RUL labels provided. The anonymization of all input variables ensures that modeling and analysis are unbiased and reproducible, without reliance on domain-specific expertise.

Table 1. Detailed Overview of the C-MAPSS Dataset Characteristics

Dataset	Train Units	Test Units	Operating Condition	Fault Modes	Sensor Variables
FD001	100	100	1	HPC Only	21
FD002	260	259	6	HPC Only	21
FD003	100	100	1	HPC + Fan	21
FD004	248	249	6	HPC +Fan	21

Constant Sensors

Some sensors exhibited little to no variance across their time series, signaling limited utility for degradation tracking and RUL prediction. A full listing of these constant or near-constant sensors by dataset is provided in Appendix A. At this stage, no sensors

were removed, in line with the project's emphasis on simplicity and full signal retention. Issues arising from non-informative variables will be addressed during later normalization, RUL capping, and feature selection steps.

Correlation Analysis – Raw Data

Pairwise correlation analysis of sensor variables identified numerous highly correlated pairs ($|r| \geq 0.80$), reflecting both redundancy and expected physical relationships within the dataset. Appendix B shows the top five highest correlated sensor pairs per dataset. Despite identifying some multicollinearity, no filtering or pruning of correlated variables was undertaken at this phase. Rather, the analysis intentionally preserved the complete sensor set to avoid prematurely discarding features that could prove valuable in a nonlinear or multivariate context.

Feature Retention Strategy

Although EDA revealed both collinearity and constant sensor patterns, all features were retained for data preprocessing and feature selection. The rationale is twofold: first, normalization and RUL capping will resolve scaling issues and address some forms of redundancy; second, a subsequent GA-based feature selection stage will empirically identify the most informative and non-redundant subset of predictors. This approach helps preserve feature combinations that might only be relevant in a multivariate or nonlinear modeling context, thereby reducing the risk of overfitting and supporting model generalizability.

Data Preprocessing

The data preprocessing pipeline is structured to maximize degradation signal fidelity and enable consistent model comparisons across various machine learning approaches.

Data Cleaning

While the C-MAPSS datasets do not contain missing values, the preprocessing workflow is designed with real-world applicability in mind. For broader relevance, missing data would be addressed via time series imputation methods, first by Kalman smoothing, followed by last-observation-carried-forward (LOCF). This approach ensures both internal gaps and trailing nulls are robustly handled by the pipeline.

Label Creation and RUL Capping

To address the inherent skewness that's presenting in the RUL target and to focus model learning on operationally relevant prediction horizons, a piecewise-linear (PWL2) labeling strategy was implemented alongside a global cap on RUL values. As shown in

Figure 2, the raw RUL distributions for each dataset exhibit substantial right-skewness, characterized by elongated upper whiskers. These extended whiskers indicate the presence of potential outlier engines with significantly longer lifespans than the median. To mitigate the disproportionate influence of these potential outliers and concentrate the modeling effort on the most common operational scenarios, a cap was set at 172 cycles, corresponding to the 75th percentile of raw RUL values across the entire datasets (highlighted by the red dashed reference line).

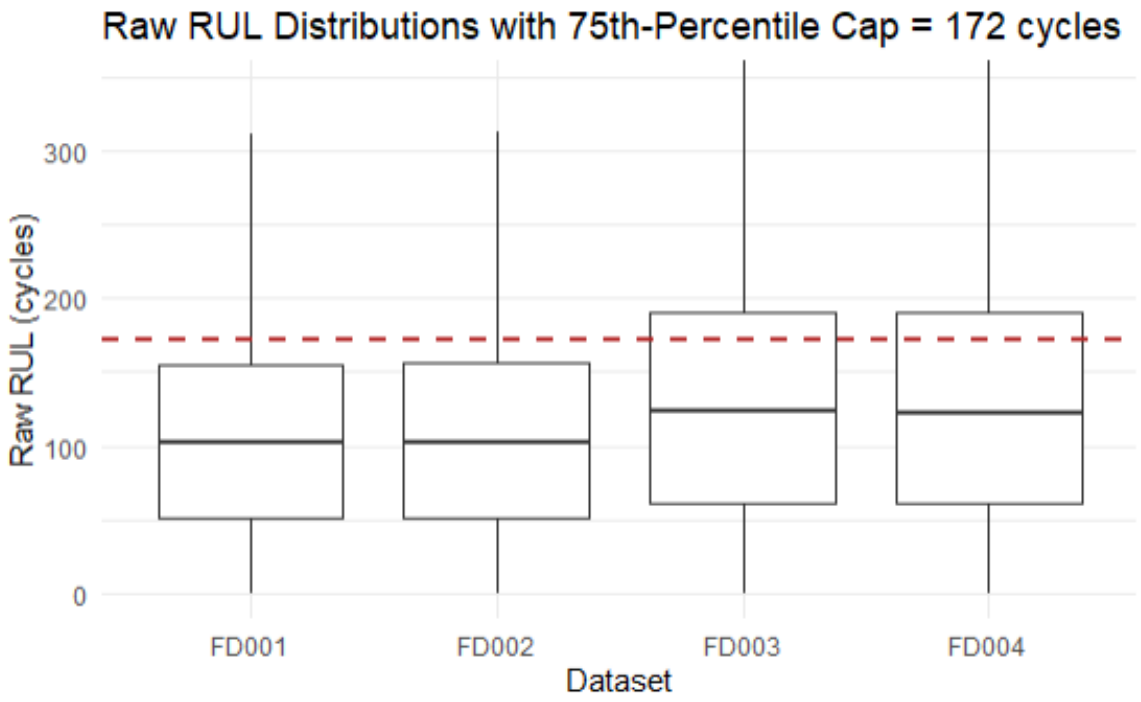

Figure 2. Raw RUL Distributions with 75th-Percentile Cap of Raw RUL

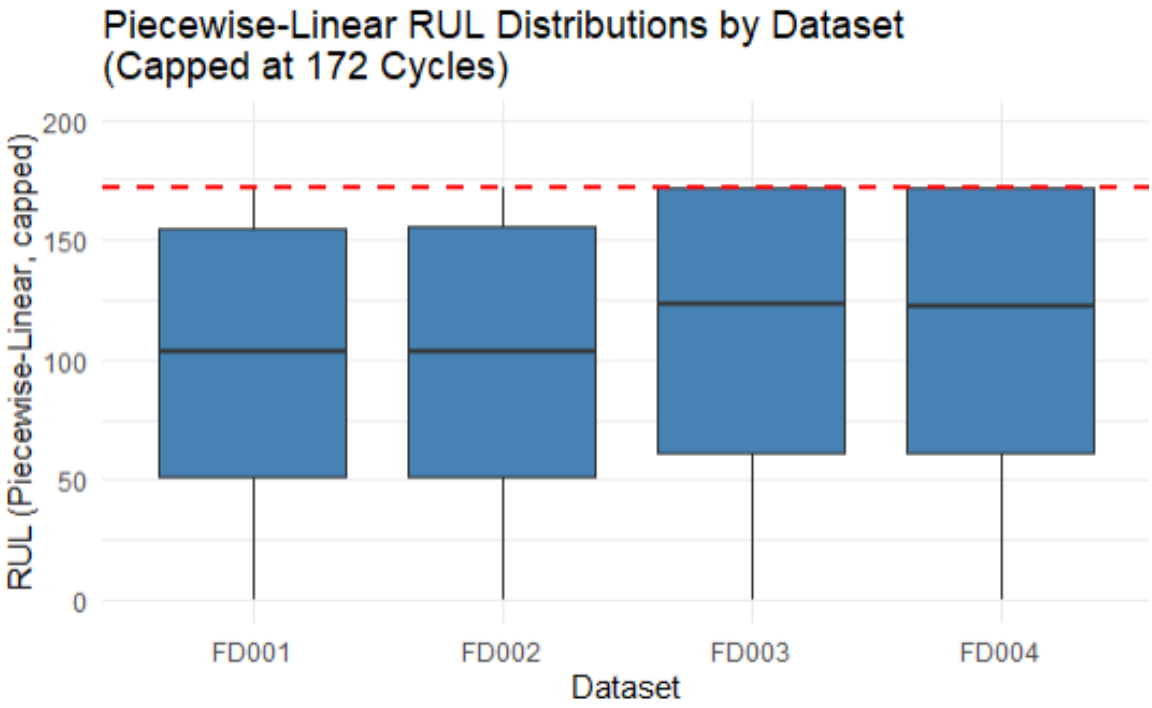

Figure 3. Piecewise-Linear (PWL2) RUL Distributions with 75th-Percentile Cap of Raw RUL

The effect of this capping and subsequent transformation is clearly visible when comparing Figures 2 and Figure 3. Figure 3 displays the distributions after applying the PWL2 strategy, which not only truncates extreme values but also compresses the dynamic range of the RUL target. This results in a more compact and less skewed distribution, thereby stabilizing the learning process and promoting improved generalization. The visual shift between the raw and PWL2-capped RUL distributions directly illustrates the impact of this preprocessing step.

By presenting both raw and capped RUL distributions, the analysis motivates the necessity of label transformation and provides empirical support for capping to ensure predictive models are tuned to the most common operational scenarios.

Data Normalization

Sensor measurements and operational settings are standardized using a "first-n-cycles" standard scaling strategy, where n = 20, meaning the first 20 operational cycles of each engine as the reference window. As shown in Table 2, normalization over the first 20 cycles successfully centers almost all operational settings and sensor signals at zero with unit variance. For FD002 and FD004, the minimum standard deviation across features is 1, indicating no constant early-cycle signals. In contrast, FD001 and FD003 exhibit sd_{min} =

0, indicating truly constant values under their specific operating and fault modes. Retaining these constant sensors at this stage preserves potentially informative signals for data-driven selection next or these may be pruned by the GA feature selector in later steps.

Table 2. Early-Cycle (1–20) Normalized Feature Ranges (μ = mean; σ = SD)

Dataset	μ_{min}	σ_{min}	μ_{max}	σ_{max}
FD001	-9.30×10^{-18}	0	9.15×10^{-18}	1
FD002	-3.78×10^{-17}	1	4.04×10^{-17}	1
FD003	-1.12×10^{-17}	0	8.93×10^{-18}	1
FD004	-1.89×10^{-17}	1	2.64×10^{-17}	1

This normalization ensures comparability across engines and features, facilitating model convergence and reducing potential biases from differing sensor ranges or initial wear states.

Feature Engineering – Sliding Time Window Processing

Rather than employing conventional rolling-window statistics, a sliding time window approach is utilized to construct sequential features from the sensor data.

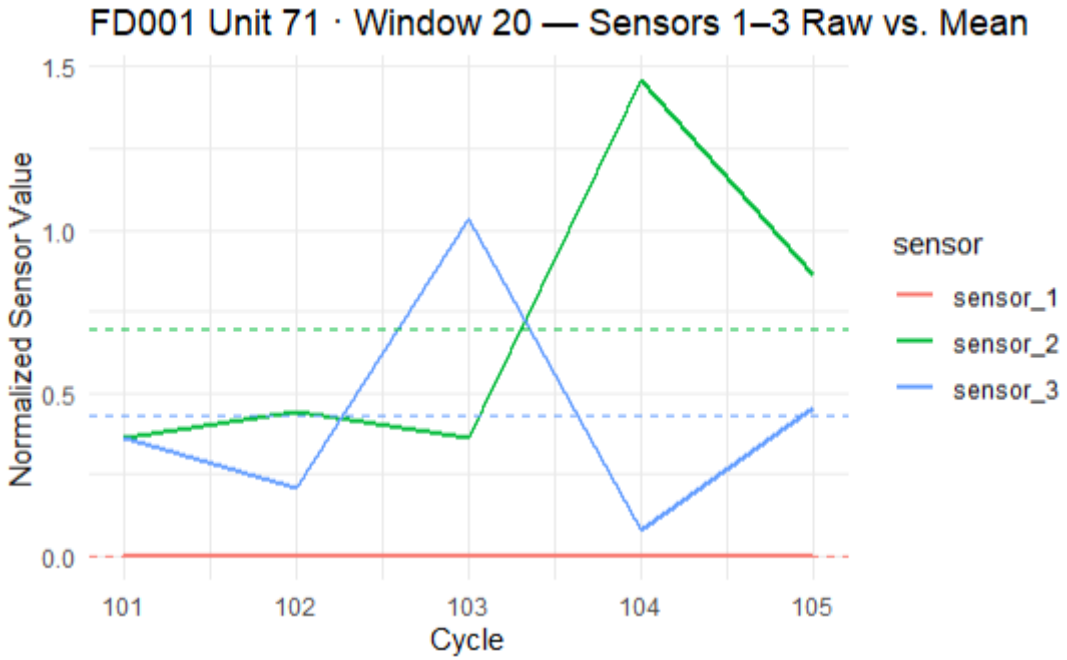

Figure 4. FD001 – Engine Unit 71 – Window 20 – Sensors 1 to 3 Raw vs. Mean

In Figure 4, Sensor 1 (shown in red) demonstrates a constant zero value, with its dashed mean line precisely overlapping the trajectory. This confirms the correct identification and handling of constant signals. In contrast, Sensors 2 and 3 display raw values that fluctuate around their respective dashed mean lines, illustrating how the sliding-window means accurately represent the local central tendency of each signal at every time step. This visual validates the effectiveness of the sliding time window approach in capturing both stationary and dynamic sensor behaviors within the dataset.

By sliding a forward-looking window of length 5–20 cycles over each time series, this strategy captures the temporal evolution of degradation more effectively and aligns with the requirements of deep learning models, such as the 1D-CNN, which are sensitive to sequence information. This method ensures that the extracted features are well suited to model the progression of engine health over time.

Correlation Analysis – Normalized Data

Table 3 lists the top eight windowed features across all sensors by absolute correlation with the PWL2 RUL target, capped at 172 cycles, using FD001 as an example.

Table 3. Top 8 Windowed Features by Absolute Correlation with PWL2-Capped RUL (FD001)

Term	**p**	**| p |**
sensor_4_mean	–0.6936	0.6936
sensor_11_mean	–0.6928	0.6928
sensor_21_mean	0.6861	0.6861
sensor_15_mean	–0.6852	0.6852
sensor_20_mean	0.6801	0.6801
sensor_17_mean	-0.6773	0.6773
sensor_4_max	–0.6756	0.6756
sensor_11_max	-0.6746	0.6746

Notably, the analysis reveals that both mean and extreme-value statistics, such as maximum, rank among the top features, indicating that hardware degradation happens through mostly gradual level shifts and occasional extreme excursions. Means of sensors 4, 11, 21, and 15 exhibit the strongest monotonic relationships with remaining useful life,

positioning them as prime candidates for GA-based feature selection. It is particularly interesting that features like sensor_21_mean and sensor_20_mean show positive correlations with degradation, underscoring that not all predictive trends align intuitively with physical wear, and highlighting the value of comprehensive empirical feature evaluation.

Principal Component Analysis (PCA) and Dimensionality Reduction

Although Principal Component Analysis (PCA) is commonly employed in prognostics for reducing dimensionality and computational cost, it fundamentally assumes that the relationships between input features are linear. As real-world engine degradation is inherently nonlinear, confirmed by an industry SME (Design Engineer, personal communication, June 10, 2025), PCA was intentionally omitted from this project's preprocessing pipeline. Instead, the complete set of 96 features of the sensor and operational feature statistics, such as Mean, SD, Min, Max, are retained. Modern machine learning algorithms and computing hardware can handle this dimensionality directly, minimizing the risk of discarding subtle yet informative degradation patterns essential for accurate RUL prediction.

While all datasets were preprocessed uniformly, subsequent model building and evaluation focused on FD001 due to compute constraints (see Limitations). Analysis of other datasets remains useful for comparative purposes and future replication.

Feature Selection

Feature selection is a critical process in RUL prediction, as it directly impacts model accuracy, interpretability, and computational efficiency. By isolating the most relevant variables, feature selection reduces the risk of overfitting and streamline downstream modeling, especially when working with high-dimensional sensor data typical of the NASA C-MAPSS benchmark. To achieve these objectives, this project adopts a parallelized GA wrapper as the core feature selection strategy, applied systematically across Ridge Regression, Random Forest, XGBoost, and 1D-CNN.

Choice of Wrapper Method

The motivation for employing a wrapper-based GA approach, rather than traditional filter methods or exhaustive searches, is due to a few advantages. First, GA can capture nonlinear feature interactions that may be overlooked by simple correlation-based or univariate filter techniques. Second, GA can efficiently search for the high-dimensional feature space without the prohibitive computational cost associated with brute-force enumeration. Third, a wrapper approach naturally aligns feature selection with model-

specific fitness, yielding a selection that is truly optimized for predictive performance. The implementation closely follows, but extends, the design proposed by Khumprom et al. (2020), with a larger population size and fewer generations, reflecting the need to limit the runtime while attempting to maintain a greater diversity.

Genetic Algorithm Configuration

In the GA framework, candidate feature subsets are encoded as binary chromosomes, with each bit representing the inclusion (1) or exclusion (0) of a particular sensor-derived feature. Fitness evaluation employs the negative RMSE, computed via three-fold CV for Ridge Regression, XGBoost, and 1D-CNN, or out-of-bag RMSE for Random Forest. GA hyperparameters were set to facilitate efficient exploration: population size = 25, maximum generations = 100, tournament selection (size = 7, representing 25% of the population), uniform crossover probability = 0.5, and mutation probability = 1 divided by the number of candidate features. Parallel computing across available cores was leveraged to enhance computational speed and efficiency.

Hyperparameter Tuning and Feature Selection

During this process, each hyperparameter combination was evaluated using grouped three-fold CV and each candidate subset was evaluated using grouped five-fold CV, keeping all data from each engine unit within the same fold to prevent data leakage. The GA systematically explored the feature space, training models with only the features specified by each candidate chromosome and selecting the subset that minimized the average RMSE. This wrapper approach was applied independently to each modeling technique to ensure that feature selection was optimally aligned with each model's learning characteristics.

Ridge Regression:

Ridge regression incorporates an L_2 penalty into the linear regression framework, effectively shrinking coefficients to address multicollinearity and reduce model variance. Hyperparameter tuning was integrated into this process, where the regularization parameter λ was determined through the previously mentioned five-fold CV method by utilizing the `cv.glmnet` package.

For FD001, the GA-driven selection process consistently favored moderate model complexity, retaining a feature subset representing over half of the total engineered predictors. Specifically, the final model for FD001 included a balanced mix of summary statistics: 15 features reflecting mean values, 11 for standard deviation, 16 for minimum, and 14 for maximum. This broad coverage indicates that both baseline operation and

typical sensor behavior (means/sd), as well as signal variability and extreme excursions (min/max), are critical for robust RUL prediction. No single summary statistic dominated the selection; rather, the model leveraged a diverse cross-section of operational settings and sensor channels. The full list of selected features is detailed in Appendix C.

Random Forest:

Random Forest is an ensemble learning method that builds multiple decision trees through bootstrap aggregation and random feature selection at each split. Hyperparameter tuning was performed using out-of-bag (OOB) errors to determine features sampled at each split (`mtry`) and number of trees (`ntree`). For FD001, the selected settings are `mtry = 32` and `ntree = 200`.

For FD001, GA selected 58 features out of the entire 96 features. The feature breakdown included 11 mean, 14 standard deviation, 18 minimum, and 15 maximum features. This selection represents about 60% of the total feature set, with a slight emphasis on minimum values, followed by substantial representation from standard deviation and maximum features. Overall, the selected subset reflects random forest's ability to leverage a balanced mix of statistical summaries and to capture predictive signals across a wide range of sensor behaviors and operational conditions. The full list of selected features is detailed in Appendix D.

XGBoost:

XGBoost, short for Extreme Gradient Boosting, is a gradient-boosted tree model that combines ensemble learning with explicit regularization to efficiently handle high-dimensional data and complex feature interactions. Hyperparameter tuning was performed using a grid search across learning rate (`eta`), tree depth (`max depth`), minimum child weight (`min_child_weight`), subsampling ratio (`subsample`), column subsampling (`colsample_bytree`), minimum loss reduction (`gamma`), L_2 regularization (`lambda`), L_1 regularization (`alpha`), and the optimal number of boosting rounds (`best_iter`). For FD001, the selected settings are `eta = 0.1`, `max_depth = 5`, `min_child_weight = 5`, `subsample = 0.95`, `colsample_bytree = 0.6`, `gamma = 0`, `lambda = 2`, `alpha = 1`, and `best_iter = 46`.

For FD001, GA selected 46 out of 96 engineered features, reflecting a 48% proportion. The feature breakdown included 14 mean, 7 standard deviation, 14 minimum, and 11 maximum features. Analysis of the selected features shows that mean and minimum statistics each make up approximately one-third of the final set, indicating a particular emphasis on central tendency and lower-bound sensor readings for RUL prediction.

Maximum and standard deviation features are also well represented, suggesting the model leverages both the variability and the extremities of sensor values. The full list of selected features is detailed in Appendix E.

One-Dimensional Convolutional Neural Networks:

The 1D-CNN model leverages convolutional neural networks' strengths in capturing temporal dependencies within raw sensor streams. As previously mentioned, a sliding time window approach is utilized as part of the feature engineering step. This windowed representation allows the 1D-CNN to directly model high-frequency sequential patterns present in the original sensor data. Hyperparameter optimization was performed via a grid search across the number of convolution filters (`filters`), convolution kernel size (`kernel`), max pooling size (`pool`), dense layer size (`dense`), learning rate (`lr`), batch size (`batch size`), and the number of training epochs (`epochs`). For FD001, the selected settings are `filters = 64`, `kernel = 3`, `pool = 3`, `dense = 32`, `learning rate = 0.001`, `batch size = 64`, and `epochs = 30`.

For FD001, GA produced a highly sparse result where only 13 out of 96 features were selected for FD001, all corresponding to raw sensor time series. No engineered summary statistics were chosen, underscoring the model's reliance on sequential, high-frequency input streams to capture the most relevant degradation signals. The full list of selected features is detailed in Appendix F.

In summary, feature selection was conducted using a parallelized GA wrapper across all modeling techniques, with model-specific CV and hyperparameter tuning to optimize predictive performance. Ridge regression and tree-based models selected a balanced subset of statistical features, while 1D-CNN focused exclusively on raw sensor streams. The GA's adaptability ensured that each model's selected features reflected its strengths in capturing degradation patterns, providing a fair foundation for subsequent training and comparative evaluation of model accuracy and robustness.

Model Building and Training

After selecting hyperparameter tuning and feature selection for each model, all models were retrained on the training dataset. Training performance was assessed using a 10-fold grouped CV per engine unit on the training dataset to estimate model performance, reported as "train RMSE". The reported "train RMSE" reflects an estimate of each model's predictive capability under the training dataset. This approach allows for consistent and fair preliminary comparisons across models prior to the final unseen test data evaluation. However, it does not necessarily capture out-of-sample performance, especially given the simplicity of FD001 and the absence of more complex scenarios in

this study. This focus on a single, less complex dataset increases the risk of model overfitting, which is addressed in later sections.

Model-Specific Training Results

Detailed performance outcomes per dataset are summarized below.

Ridge Regression:

Trained with optimal λ via nested cross-validation; 10-fold CV results are summarized in Table 4.

Table 4. Train RMSE for Ridge Regression

Dataset	Train RMSE	Standard Deviation
FD001	31.77	3.53

Random Forest:

Utilized best `mtry/ntree` from OOB optimization; 10-fold CV results are summarized in Table 5.

Table 5. Train RMSE for Random Forest

Dataset	Train RMSE	Standard Deviation
FD001	30.07	3.17

XGBoost:

Evaluated with tuned hyperparameters; 10-fold CV results are summarized in Table 6.

Table 6. Train RMSE for XGBoost

Dataset	Train RMSE	Standard Deviation
FD001	30.30	3.58

1D-CNN:

Assessed with best window and network parameters; 10-fold CV results are summarized in Table 7.

Table 7. Train RMSE for 1D-CNN

Dataset	Train RMSE	Standard Deviation
FD001	69.47	4.74

Ridge regression, random forest, and XGBoost all demonstrated strong predictive performance (Train RMSE ≈ 30–31), with 1D-CNN performing significantly worse with a Train RMSE at 69.47. However, the actual evaluation of the model performance depends on the prediction results from the test data, compared to the true RUL values from the C-MAPSS dataset.

FINDINGS

Model Testing

For all models, test set performance was evaluated using predictions from the final window of each engine, corresponding to the true RUL at the end of each unit's operational cycle. After retraining each model on the complete training set, we assessed final performance on the held-out test data using two complementary metrics: RMSE Capped and RMSE True. RMSE Capped quantifies the error of model predictions against PWL2-capped RUL labels across all sliding windows, ensuring alignment with the training objective. RMSE True measures the error between each engine's last-window prediction and its actual cycles-to-failure from the original RUL file, thereby capturing end-of-life predictive accuracy. This dual-metric evaluation is consistent with best practices in prognostics research (Wang et al., 2025; Li et al., 2024), and mirrors real-world maintenance scenarios where only the most recent prediction informs operational decisions.

Model-Specific Test Results

Detailed performance outcomes per model are summarized in Table 8.

Table 8. RMSE Capped and RMSE Raw for FD001 across all models

Dataset	Model	RMSE Capped	RMSE True
FD001	Ridge	74.24	36.32
FD001	Random Forest	73.03	34.71
FD001	XGBoost	73.45	34.15
FD001	CNN	56.57	52.51

Ridge regression, random forest, and XGBoost all achieve comparable end-of-life RMSE True values (ranging from 34.15 to 36.32), while 1D-CNN lags at 52.41. However, RMSE Capped scores are substantially higher for all models, spanning from 56.57 for 1D-CNN to 74.24 for ridge regression, confirming that average error is larger when evaluated across all sliding windows, particularly during the early plateau phase shaped by RUL capping. Notably, XGBoost outperforms the other machine learning models on end-of-life accuracy, aligning with findings by Safavi et al. (2024) that a well-tuned XGBoost model can outperform deep learning models for RUL prediction.

The 1D-CNN model achieved 52.51 for its RMSE True values on the FD001 dataset, while Train RMSE is higher at 69.47. This result indicates that the model generalizes better to the end-of-life prediction scenario than its average performance across all training windows would suggest. However, compared to other models, the 1D-CNN model does not demonstrate a performance advantage in this less complex FD001 dataset, potentially due to a limited hyperparameter search and the use of early stopping with a short patience threshold (`patience = 5`), which may have prematurely terminated training before the network could recover from initial overfitting.

Additionally, ridge regression, random forest, and XGBoost all exhibit Train RMSE values that are more than one standard deviation lower than their respective RMSE on the true test set, indicating a degree of overfitting. This is not unexpected, given that FD001 is the simplest scenario among the C-MAPSS datasets, featuring only a single operating condition and a single fault mode. In contrast, the other datasets present increased complexity through multiple operating conditions and fault types, making overfitting less likely in those cases.

Because only FD001 has been evaluated, the standard deviation of RMSE across RMSE True cannot be reported at this stage, as there is no cross-dataset variability available. In future work, we plan to extend the scope to include FD002 through FD004, which will enable mean and standard deviation summaries of RMSE and provide a more comprehensive assessment of performance consistency across datasets. Nonetheless, the current hold-out evaluation framework offers a clear and unbiased basis for comparing model performance on both capped and true RUL prediction in this proof-of-concept analysis.

LIMITATIONS AND FUTURE WORK

This study faced several key limitations that constrained the scope of analysis and impacted on the depth of exploration originally intended.

First, significant challenges arose during the initial setup and configuration of GPU-accelerated deep learning frameworks. Despite extensive troubleshooting and validation efforts, including verifying hardware compatibility (NVIDIA RTX 3060 GPU), CUDA toolkit installation, and environment configurations, persistent dependency conflicts and dynamic link library (DLL) initialization errors prevented successful integration of GPU computation within the RStudio, reticulate, and Conda environment on Windows. These GPU-related issues substantially delayed the early stages of the project, causing internal deadlines to be missed and ultimately limiting computational resources to CPU-only parallelism. Consequently, the planned deep learning model, the Bidirectional Long Short-Term Memory (BiLSTM), had to be dropped from the analysis scope due to its significant computational resource requirements.

Subsequently, before the project is moving onto the 1D-CNN model for Model Building and Training phase and we had a chance to document the available findings, a critical software stability issue occurred on July 3, 2025, the "white screen of death" rendering failure, necessitating a complete restart of the machine and prompting a re-evaluation of the project's scope. Given the remaining time available, the analysis was first scaled down from the original plan of utilizing all four NASA C-MAPSS datasets (FD001–FD004) to just two (FD001 and FD004) with a reduced hyperparameter grid. However, even with this reduced scope, computational bottlenecks and excessively long processing times associated specifically with dataset FD004, mostly during GA-based feature selection for XGBoost, proved insurmountable within the remaining timeframe. Ultimately, these constraints compelled a further reduction of scope to include only FD001.

Furthermore, several methodological simplifications were employed due to the constrained timeline. For data normalization, the "first-n-cycle" standard scaling approach was utilized with a fixed parameter (n=20), without employing methods such as the elbow criterion or CV to identify an optimal choice of this parameter. While pragmatic, this decision may introduce potential biases or suboptimal scaling effects.

Lastly, while significant effort was invested into reviewing existing literature, the sheer breadth and diversity of available research on predictive maintenance and machinery prognostics means that complete coverage is impractical. Consequently, it is inevitable that relevant works, particularly those published in languages other than English, were omitted from consideration, potentially limiting the contextual comprehensiveness of this research.

Looking forward, addressing these limitations is a key priority. Future work will revisit GPU-accelerated deep learning implementations in a more predictable and stable

computing environment, such as a Linux-based virtual environment or a newer NVIDIA GPU, to better leverage high-performance computing resources. Finally, the project scope will be expanded to encompass all four NASA C-MAPSS datasets, enabling a comprehensive evaluation of modeling techniques across diverse operational conditions.

CONCLUSION

In conclusion, this project sets out to benchmark the predictive performance of traditional and advanced machine learning models for estimating the RUL of turbofan engines using the NASA C-MAPSS dataset. Although the initial scope included all four C-MAPSS datasets, severe computational and timeline constraints ultimately required narrowing the analysis to FD001 for this proof-of-concept evaluation.

Among the models compared, XGBoost achieved the lowest error for end-of-life RUL prediction, with ridge regression and random forest showing nearly equivalent performance. While all models demonstrated higher average error when evaluated across all operational windows, reflecting the influence of RUL capping during early engine life, tree-based methods consistently outperformed the deep learning approach in this context. The 1D-CNN model, while generalizing better to the test scenario than its overall training performance, did not provide a significant advantage on this relatively simple dataset. This work highlights the practical utility of a model-agnostic GA for feature selection, supporting fair comparison and rigorous benchmarking across a range of modeling strategies.

A key reflection from this work is the importance of rapid prototyping and iterative development when building data science workflows for real-world scenarios, as opposed to a strictly sequential approach typical in academic settings. Prioritizing a working proof-of-concept early would have accelerated troubleshooting and allowed more rapid adaptation to the inevitable challenges of real-world data science projects.

Future research should prioritize validating this pipeline across all four C-MAPSS datasets and under a broader range of operational and fault scenarios. Such extensions will be critical for robustly benchmarking model performance and for developing generalizable, resource-efficient predictive maintenance solutions.

REFERENCES

1. Asif, O., Haider, S. A., Naqvi, S. R., Zaki, J. F. W., Kwak, K.-S., & Islam, S. M. R. (2022). A deep learning model for remaining useful life prediction of aircraft

turbofan engine on C-MAPSS dataset. IEEE Access, 10, 95425–95440. https://doi.org/10.1109/ACCESS.2022.3203406

2. Ensarioğlu, K., İnkaya, T., & Emel, E. (2023). Remaining useful life estimation of turbofan engines with deep learning using change-point detection based labeling and feature engineering. Applied Sciences, 13(21), 11893. https://doi.org/10.3390/app132111893
3. Ferreira, C., & Gonçalves, G. (2022). Remaining useful life prediction and challenges: A literature review on the use of machine learning methods. Journal of Manufacturing Systems, 63, 550–562. https://doi.org/10.1016/j.jmsy.2022.05.010
4. Hu, C., Youn, B. D., Wang, P., & Yoon, J. T. (2012). Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life. Reliability Engineering & System Safety, 103, 120–135. https://doi.org/10.1016/j.ress.2012.03.008
5. Khumprom, P., Grewell, D., & Yodo, N. (2020). Deep neural network feature selection approaches for data-driven prognostic model of aircraft engines. Aerospace, 7(9), 132. https://doi.org/10.3390/aerospace7090132
6. Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. (2018). Machinery health prognostics: A systematic review from data acquisition to RUL prediction. Mechanical Systems and Signal Processing, 104, 799–834. https://doi.org/10.1016/j.ymssp.2017.11.016
7. Li, F., Dai, Z., Jiang, L., Song, C., Zhong, C., & Chen, Y. (2024). Prediction of the remaining useful life of bearings through CNN-Bi-LSTM-based domain adaptation model. Sensors, 24, 6906. https://doi.org/10.3390/s24216906
8. Li, X., Zhang, W., Ding, Y., Cai, J., & Yan, X. (2025). Rolling bearing degradation stage division and RUL prediction based on recursive exponential slow feature analysis and Bi-LSTM model. Reliability Engineering & System Safety, 252, 110923. https://doi.org/10.1016/j.ress.2025.110923
9. Ramasso, E., & Saxena, A. (2020). Performance benchmarking and analysis of prognostic methods for CMAPSS datasets. International Journal of Prognostics and Health Management, 5(2). https://doi.org/10.36001/ijphm.2014.v5i2.2236
10. Safavi, S., Abolhasani, A., & Gordi, M. (2024). Early prediction of battery remaining useful life using a CNN-XGBoost model and Coati optimisation. Journal of Energy Storage, 65, 550–562. http://dx.doi.org/10.1016/j.est.2024.113176
11. Safavi, V., Mohammadi Vaniar, A., Bazmohammadi, N., Vasquez, J. C., & Guerrero, J. M. (2024). Battery remaining useful life prediction using machine learning models: A comparative study. Information, 15, 124. https://doi.org/10.3390/info15030124
12. Vafaie, H., & Imam, I. (1994). Feature selection methods: Genetic algorithms vs. greedy-like search. https://www.mli.gmu.edu/papers/91-95/94-10.pdf

13. Wang, F., Liu, A., Qu, C., Xiong, R., & Chen, L. (2025). A deep-learning method for remaining useful life prediction of power machinery via dual-attention mechanism. Sensors, 25, 497. https://doi.org/10.3390/s25020497
14. Wang, H., Li, D., Li, D., Liu, C., Yang, X., & Zhu, G. (2023). Remaining useful life prediction of aircraft turbofan engine based on random forest feature selection and multi-layer perceptron. Applied Sciences, 13(12), 7186. https://doi.org/10.3390/app13127186
15. Wu, F., Wu, Q., Tan, Y., & Xu, X. (2024). Remaining useful life prediction based on deep learning: A survey. Sensors, 24(11), 3454. https://doi.org/10.3390/s24113454

APPENDICES

Appendix A: Constant/Near-Constant Sensors Identified by Dataset

Dataset	Constant/Near-Constant Sensors
FD001	senor_1, senor_5, senor_10, senor_16, senor_18, senor_19
FD002	senor_16
FD003	senor_1, senor_5, senor_16, senor_18, senor_19
FD004	senor_16

Appendix B: Top Five Highest Correlated Sensor Pairs per Dataset

Dataset	Sensor Pair	Correlation (r)
FD001	sensor_9 ↔ sensor_14	+0.96
	sensor_11 ↔ sensor_12	-0.85
	sensor_4 ↔ sensor_11	+0.83
	sensor_8 ↔ sensor_13	+0.83
	sensor_7 ↔ sensor_11	-0.82
FD002	sensor_5 ↔ sensor_6	+1.00
	sensor_7 ↔ sensor_12	+1.00

	sensor_12 ↔ sensor_20	+1.00
	sensor_12 ↔ sensor_21	+1.00
	sensor_20 ↔ sensor_21	+1.00
FD003	sensor_7 ↔ sensor_12	+0.99
	sensor_8 ↔ sensor_13	+0.96
	sensor_9 ↔ sensor_14	+0.95
	sensor_4 ↔ sensor_11	+0.85
	sensor_7 ↔ sensor_10	+0.83
FD004	sensor_5 ↔ sensor_6	+1.00
	sensor_7 ↔ sensor_12	+1.00
	sensor_12 ↔ sensor_20	+1.00
	sensor_12 ↔ sensor_21	+1.00
	sensor_20 ↔ sensor_21	+1.00

Appendix C: Ridge Regression (FD001): Selected Features by Type

Type	**Features**
Mean	op_setting_1_mean, op_setting_1_min, op_setting_3_mean, sensor_1_mean, sensor_2_mean, sensor_3_mean, sensor_4_mean, sensor_5_mean, sensor_6_max, sensor_7_mean, sensor_9_mean, sensor_10_mean, sensor_11_mean, sensor_12_mean, sensor_14_mean, sensor_15_mean, sensor_16_min, sensor_17_mean, sensor_18_min, sensor_19_min, sensor_20_mean, sensor_20_min, sensor_21_mean
SD	op_setting_3_sd, sensor_1_sd, sensor_3_sd, sensor_5_sd, sensor_8_sd, sensor_9_sd, sensor_13_sd, sensor_17_sd, sensor_18_sd, sensor_20_sd, sensor_21_sd
Min	op_setting_1_min, op_setting_3_min, sensor_1_min, sensor_3_min, sensor_4_min, sensor_5_min, sensor_7_min, sensor_9_min, sensor_10_min, sensor_11_min, sensor_12_min, sensor_13_min, sensor_14_min, sensor_16_min, sensor_18_min, sensor_19_min, sensor_20_min,

	sensor_21_min
Max	op_setting_3_max, sensor_1_max, sensor_3_max, sensor_4_max, sensor_5_max, sensor_7_max, sensor_8_max, sensor_9_max, sensor_10_max, sensor_11_max, sensor_12_max, sensor_13_max, sensor_14_max, sensor_15_max, sensor_16_max, sensor_17_max, sensor_18_max, sensor_19_max, sensor_20_max, sensor_21_max

Appendix D: Random Forest (FD001): Selected Features by Type

Type	**Features**
Mean	op_setting_1_mean, op_setting_2_mean, sensor_1_mean, sensor_2_mean, sensor_6_mean, sensor_7_mean, sensor_8_mean, sensor_10_mean, sensor_16_mean, sensor_17_mean, sensor_18_mean, sensor_19_mean, sensor_20_mean
SD	sensor_1_sd, sensor_2_sd, sensor_3_sd, sensor_5_sd, sensor_7_sd, sensor_8_sd, sensor_9_sd, sensor_10_sd, sensor_11_sd, sensor_12_sd, sensor_14_sd, sensor_15_sd, sensor_16_sd, sensor_17_sd, sensor_18_sd
Min	op_setting_1_min, op_setting_2_min, sensor_1_min, sensor_5_min, sensor_6_min, sensor_7_min, sensor_8_min, sensor_12_min, sensor_13_min, sensor_14_min, sensor_15_min, sensor_18_min, sensor_19_min, sensor_21_min
Max	op_setting_1_max, op_setting_2_max, sensor_1_max, sensor_3_max, sensor_4_max, sensor_8_max, sensor_9_max, sensor_10_max, sensor_12_max, sensor_13_max, sensor_14_max, sensor_15_max, sensor_16_max, sensor_18_max, sensor_20_max

Appendix E: XGBoost (FD001): Selected Features by Type

Type	**Features**
Mean	op_setting_1_mean, op_setting_2_mean, sensor_3_mean, sensor_4_mean, sensor_6_mean, sensor_8_mean, sensor_9_mean, sensor_11_mean, sensor_12_mean, sensor_13_mean, sensor_14_mean, sensor_15_mean, sensor_17_mean, sensor_18_mean, sensor_19_mean, sensor_20_mean
SD	op_setting_1_sd, sensor_2_sd, sensor_4_sd, sensor_8_sd, sensor_9_sd,

	sensor_10_sd, sensor_17_sd
Min	op_setting_2_min, op_setting_3_min, sensor_1_min, sensor_2_min, sensor_4_min, sensor_6_min, sensor_8_min, sensor_9_min, sensor_10_min, sensor_11_min, sensor_12_min, sensor_15_min, sensor_21_min
Max	op_setting_2_max, op_setting_3_max, sensor_8_max, sensor_9_max, sensor_10_max, sensor_13_max, sensor_14_max, sensor_17_max, sensor_20_max, sensor_21_max

Appendix F: 1D-CNN (FD001): Selected Features by Type

Type	**Features**
Raw Sensor	sensor_2, sensor_4, sensor_5, sensor_8, sensor_9, sensor_10, sensor_11, sensor_14, sensor_15, sensor_17, sensor_18, sensor_19, sensor_21